Incremental Evolution Strategy for Function Optimization

[1]Sheng-Uei Guan and [2]Wenting Mo
[1]School of Engineering and Design
Brunel University, Uxbridge, Middlesex, UB8 3PH, UK

[2]Department of Electrical and Computer Engineering
National University of Singapore
10 Kent Ridge Crescent, Singapore 119260

Abstract

This paper presents a novel evolutionary approach for function optimization — Incremental Evolution Strategy (IES). Two strategies are proposed. One is to evolve the input variables incrementally. The whole evolution consists of several phases and one more variable is focused in each phase. The number of phases is equal to the number of variables in maximum. Each phase is composed of two stages: in the single-variable evolution (SVE) stage, evolution is taken on one independent variable in a series of cutting planes; in the multi-variable evolving (MVE) stage, the initial population is formed by integrating the populations obtained by the SVE and the MVE in the last phase. And the evolution is taken on the incremented variable set. The other strategy is a hybrid of particle swarm optimization (PSO) and evolution strategy (ES). PSO is applied to adjust the cutting planes/hyper-planes (in SVEs/MVEs) while (1+1)-ES is applied to searching optima in the cutting planes/hyper-planes. The results of experiments show that the performance of IES is generally better than that of three other evolutionary algorithms, improved normal GA, PSO and SADE_CERAF, in the sense that IES finds solutions closer to the true optima and with more optimal objective values.

Keywords: evolution strategy, function optimization, incremental evolution, particle swarm optimization.

1. INTRODUCTION

1.1 Background

The need to solve function optimization problems arises in one form or another in the engineering world. Although many optimization techniques have been developed, there are still large classes of functions which are beyond the reach of analytical methods and present significant difficulties for numerical techniques. Unfortunately, such functions are quite commonplace, for example, functions which are not continuous or differentiable everywhere, functions which are non-convex, multi-modal (multiple peaks), and functions which contain noise. As a consequence, there is continuing search for new and more robust optimization techniques capable of handling such problems. In the past few decades we have seen an increasing interest in biologically motivated approaches of solving optimization problems, including neural networks (NNs), evolutionary algorithms (EAs), and particle swarm optimization (PSO) [23-25].

Evolutionary Algorithms (EAs) serve as popular tools for search, optimization, machine learning and solving design problems. Historically, genetic algorithms (GAs) and evolution strategies (ESs) are two of the most basic forms of EAs. Both of them have been used for optimization. GAs have long been viewed as multi-purpose tools with applications in search, optimization, design and machine learning [5, 6], while most of the work in ESs focused on optimization [7-9].

Evolution Strategies were developed in Germany under the leadership of Ingo Rechenberg and Hans-Paul Schwefel [25]. ESs tend to use more direct representations than GAs [4], thus they are generally applied to real-valued representations of optimization problems. And in ESs mutation is emphasized over recombination. The two basic types of ESs are known as the (μ,λ)-ES and the $(\mu+\lambda)$-ES (μ is the size of the parent population and λ is the number of

offspring that are produced in a single generation before selection is applied). In a (μ,λ)-ES the offspring replace the parents. In a $(\mu+\lambda)$-ES, selection picks from both the offspring and the parents to create the next generation [10]. Different values of parameter μ and λ could have a large impact on the performance of ESs. In this paper, $(1+1)$-ES is chosen for the algorithm proposed for its simplicity using [10].

Particle swarm optimization (PSO) is a novel multi-agent optimization system (MAOS) inspired by social behavior metaphor [11]. And the concept of a more-or-less permanent social topology is fundamental to PSO [11, 18]. Each agent in PSO, called *particle*, flies in a d-dimensional space S according to the historical experiences of its own and its colleagues. The velocity and location for the ith particle is represented as $\overrightarrow{v_i}=(v_{i1},\ldots,v_{ij},\ldots,v_{id})$ and $\overrightarrow{x_i}=(x_{i1},\ldots,x_{ij},\ldots,x_{id})$, respectively. Its best previous location is recorded and represented as $\overrightarrow{p_i}=(p_{i1},\ldots,p_{ij},\ldots,p_{id})$, which is also called *pbest*. The index of the best *pbest* is represented by the symbol g, and is called *gbest*. At each step, the particles are manipulated according to the following equations [12]:

$$v_{ij}=w\cdot v_{ij}+c_1\cdot rand(\ )\cdot(p_{ij}-x_{ij})+c_2\cdot rand(\ )\cdot(p_{gj}-x_{ij})$$
$$x_{ij}=x_{ij}+v_{ij}$$

where w is inertia weight, c_1 and c_2 are acceleration constants between 0 and 1, $rand(\ )$ represent random values between 0 and 1.

Several researchers have analyzed this optimization algorithm empirically [13-15] and theoretically [16, 17]. They showed that the particles oscillate in different sinusoidal waves and converge quickly, especially for PSO with a small w [15] or constriction coefficient [16].

1.2 Challenges and proposed solution

Although biologically inspired algorithms are more effective for the difficult functions discussed above than some classical numerical methods, some features of the functions, such as ridges and local optima, often obstruct them from converging to the global optima. The algorithmic challenge in handling ridges is to change multiple variables simultaneously in order to search in the direction of the ridge orientation and thereby avoid reduction in fitness. A lot of problems with ridges could be successfully solved by self-adaptive ES [7]. However, self-adaptive ES is unsuitable for high-dimensional problems. According to [10], a chromosome of self-adaptive ES should include the "object parameter" together with the "strategy parameters" and even with the "rotation parameter". Thus if there are d object parameters, namely, d variables in the objective function, there will be d strategy parameters and $d(d-1)/2$ rotation parameters. When the problem is a high dimensional one, that's to say d is a large value, the chromosome will become quite complex and the performance of ES may be not so satisfying. For instance, a 100-dimensional problem requires 4950 angles to allow rotation between all dimensions. Thus we attempt to focus on evolving in lower dimensions, that's why the incremental mechanism is used. Problem with local optima, namely multi-modal problem, is also quite common and unavoidable in real-world applications. And it is well received to be hard to handle, especially when the number of local optima is rather large. The basic algorithms have a tendency to stagnate in a local optimum because escaping from such optima may require a significant amount of backtracking, namely "downhill movement", before new fitness improvements occur. Thus, a great deal of work has been dedicated to it. However, since the dimensionality of functions they tackled is not very high, usually less than 30, the number of local optima won't be huge then [29].

In this paper a new algorithm, Incremental Evolution Strategy (IES), is proposed, To reduce the dimensionality of searching, we slice the searching space with cutting planes/hyper-planes. Particle swarm optimization is used to adjust the cutting

planes/hyper-planes to approach the planes/hyper-planes containing global optima while ES is used to search the optima in these cutting planes/hyper-planes. In fact, some researchers did try to combine the power of EAs and PSO. In [19], a hybrid algorithm of GA and PSO is proposed and the results showed that this algorithm outperforms simple PSO and simple GA. However, this PSO-GA hybrid is just a simple combination of the two algorithms, which is done by taking the population of PSO when the improvement starts to level off and using it as the starting population of GA.

IES uses incremental evolution strategy as a basic vehicle for optimizing, together with particle swarm optimization to assist in incremental evolving. In fact, the concept of incremental learning/evolution has been proved feasible and effective in some previous work of our team, including incremental learning both in the input space and the output space [27-32] and incrementally evolving multi-objective problems [33]. Different from normal ESs which evolve variables in their full dimension to optimize the objective function, IES evolves the variables one after another under a scenario of continuous incremental optimization. Each time when a new variable is introduced, single-variable evolution is first implemented under particle swarm (PS) assisted ES, then the found solutions are integrated with the solutions found earlier, and lastly multi-variable evolution is implemented with regard to the variable set evolved so far. When the dimensionality is high, not all the variables need to be evolved individually, rather, a stagnation criterion will decide how many variables need individual evolution. The simulation results showed that IES can achieve generally better performance than normal ES in terms of obtaining solutions with higher quality both in the input space and the output space.

For performance comparison, SADE_CERAF, an evolutionary algorithm claimed for global optimization in real domains, is used. Simplified Atavistic Differential Evolution (SADE) combines the features of differential evolution (DE) with those of traditional genetic algorithms [21]. DE is a modern and efficient optimization method

essentially relying on so-called differential operator, which was invented as the solution method for the Chebychev trial polynomial problem by Stone and Price [20]. CERAF is a technology enabling the population of solutions to escape from local extremes, which aims to improve the performance of SADE by preventing premature convergence [21].

The rest of the paper is organized as follows. In section 2, orthographic projection of objective function is presented and cutting plane mechanism is defined. Based on the analysis in section 2, section 3 proposes the new algorithm, IES. Section 4 presents the results of experiments and relevant analysis. Section 5 discusses why IES works. Section 6 concludes this paper.

2. RELATED THEORY

The concept of IES originates from the idea of function projection. This section presents the concept of function projection and its extension.

2.1 Motivation

As we know, a 3-dimensional object can be described exactly by three-view orthographic projection drawing, which is some kind of mechanical drawing. A three-view orthographic projection drawing shows the front, top, and right sides of an object as shown in Fig. 1. An important factor in a three-view drawing is the relationship among height, width, and depth. The top and front views share width. The top and side views share depth. The front and side views share height.

With respect to a function optimization problem, the aim is to find the optimal objective value of a function with d variables. Such a function can be seen as a hyper-surface in the $(d+1)$-dimensional space and the nadir (or the zenith in maximization problem) of the hyper-surface is to be found. (Since any maximization

problem can be turned into a minimization one, this paper considers minimization only.) Inspired by the phenomenon that in a three-view orthographic projection drawing the height information won't be lost with orthographic projection from the front view or the side view, we consider taking orthographic projection of objective function from 'variable view', which means to project the corresponding hyper-surface orthographically onto variable-objective value planes. The detailed formal descriptions of this concept are presented as follows.

2.2 Orthographic Projection of Function

Consider a single objective minimization problem with d attributes in the input space, we formulate the optimization problem as finding $X=(x_1,x_2,\cdots,x_d)$ to minimize the value of $y=f(X)$ within the feasible input region I.

Definitions:

1) Feasible input region I is the set of all vectors that satisfy the constraints and bounds of the problem.
2) $\{\overrightarrow{u_1},\overrightarrow{u_2},\cdots,\overrightarrow{u_d},\overrightarrow{u_{d+1}}\}$ is the set of orthogonal bases in the $(d+1)$-dimensional space R^{d+1}, corresponding to $\{x_1,x_2,\cdots,x_d,y\}$. A 3-dimensional example is demonstrated as follows.
3) Orthographic projection refers to the projection along the orthogonal bases, vectors $\overrightarrow{u_i}$, $i\in 1,2,\ldots,d+1$.
4) P_{x_i-y}, $i=1,2,\cdots,d$, is the boundary of the orthographic projection of the original function $y=f(x_1,x_2,\cdots,x_d)$ on the x_i-y plane. And we use a function $y^{(i)}=f^{(i)}(x_i)$ to describe P_{x_i-y}.
5) To facilitate discussion without losing generality, assume that there is only one global optimal solution $(x_1^{\,g},x_2^{\,g},\cdots,x_d^{\,g},y^g)$ for the original problem. And one

optimal solution $\left(x_i^*, y^{(i)*}\right)$ for each projected problem could be found.

In IES, we try to minimize the problem incrementally. Consider the extreme situation, in which the variables are incrementally evolved one by one. That means we project the original problem into d projected sub-problems, i.e. to find:
$\left\{\left(x_1^*, y^{(1)*}\right), \left(x_2^*, y^{(2)*}\right), \cdots, \left(x_d^*, y^{(d)*}\right)\right\}$.

Statement 1: The minimum of $P_{x_i - y}$, $\left(x_i^*, y^{(i)*}\right)$, is the projection of the global minimum $\left(x_1^g, x_2^g, \cdots, x_d^g, y^g\right)$ of the original problem on the $x_i - y$ plane, $i = 1, 2, \cdots, d$.

Apagoge is used:
Assume $\left(x_i^*, y^{(i)*}\right)$ is not the projection of $\left(x_1^g, x_2^g, \cdots, x_d^g, y^g\right)$ on the $x_i - y$ plane, $i = 1, 2, \cdots, d$.

When we project $y = f\left(x_1, x_2, \cdots, x_d\right)$ onto the $x_i - y$ plane, each point $(x_1^o, x_2^o, \cdots, x_d^o, y^o)$ on the original hyper-surface $y = f\left(x_1, x_2, \cdots, x_d\right)$ corresponds to a point in the area of the orthographic projection. And, when taking orthographic projection along all the directions except $\vec{u_i}$ and $\overrightarrow{u_{d+1}}$ in the R^{d+1} space, there is $\vec{u_j} \times \vec{u_j} = 0$, $j = 1, 2, \ldots, d$, $j \neq i$. And in the $x_i - y$ plane, there is $\vec{u_i} \bullet \vec{u_i} = 1$ and $\overrightarrow{u_{d+1}} \bullet \overrightarrow{u_{d+1}} = 1$. Thus the projection point $\left(x_i^p, y^{(i)p}\right)$ will be $\begin{cases} x_i^p = x_i^o \\ y^{(i)p} = y^o \end{cases}$. That's to say, the "height" information in the ith and the $d+1$th dimensions is retained, and the "height" information in the other dimensions is discarded.

So there is at least one point on the original hyper-surface corresponding to each point $\left(x_i, y^{(i)}\right)$ on $P_{x_i - y}$, a curve described by $y^{(i)} = f^{(i)}(x_i)$.

According to the assumption, there should be a point $(x_i^c, y^{(i)c})$ in the area of the projection in the $x_i - y$ plane other than $\left(x_i^*, y^{(i)*}\right)$ corresponding to the global minimum $\left(x_1^g, x_2^g, \cdots, x_d^g, y^g\right)$, which has $\begin{cases} x_i^c = x_i^g \\ y^{(i)c} = y^g \end{cases}$.

With regard to the global minimum, $y^g = \min\left\{y^o\right\}$, then it can be deduced that $y^{(i)c} = \min\left\{y^{(i)p}\right\}$, that's to say $y^{(i)c} < y^{(i)*}$. However, there is the premise that $\left(x_i^*, y^{(i)*}\right)$ is the minimum point of $P_{x_i - y}$, and since $P_{x_i - y}$ is the boundary of projection on the $x_i - y$ plane, $\left(x_i^*, y^{(i)*}\right)$ is the minimum point of the area of projection.

Therefore, the conclusion contradicts with the premise, and it is proved that $\left(x_i^*, y^{(i)*}\right)$ is the projection of the global minimum point on the $x_i - y$ plane.

In some cases the assumption that there is only one global optimal solution of the original problem may not hold. This means there could be multiple global optimal solutions of the original problem. Nonetheless, it is obvious that the proof above still holds in the sense that the global optimal solutions won't lose their predominance in terms of y value after taking orthographic projection. The only difference would be that multiple optimal solutions could be found in some or all of the projected problems, which are the projections of the original global optimal solutions. Obviously the conclusion can be generalized to orthographic projection of higher dimensions. That is: the minima of the boundary functions resulting from the projection of the hyper-surface corresponding to the original objective function, are the projection of the global optima.

2.3 Cutting Plane Mechanism: Local via Global Search

From the previous discussions, it is proved that if the exact boundary functions of the orthographic projection can be obtained. We can find the optimum easily by just

evolving a one-variable problem. Unfortunately, we cannot easily find the exact function which describes the projection boundaries. Nevertheless, we can still make use of the features and concepts discussed above through a mechanism called cutting plane.

Definitions:

1) For an optimization problem with d variables, fix some variables and evolve the other variables. The fixed ones are called *unconcerned variable*s (X^{uc}) while the rest are called *concerned variable*s (X^{c}).
2) A point in the space of concerned variables is called the *projection* of the corresponding point in the original space and that the original point is the *proto-point* of the projection.
3) When there is only one concerned variable, the projection method is called *cutting plane mechanism*. The concerned variable-objective value plane is the *cutting plane*. And the cutting plane intersects the original hyper-surface resulting in an *intercepted curve*.
4) If the fixed values for those unconcerned variables are equal to the values of corresponding variables of the global optimum, i.e. $\left|x_i^{uc} - x_i^{g}\right| = 0$ (where $x_i^{uc} \in X^{uc}$, x_i^{g} is the value of corresponding variable of the global optimum), the cutting plane is called the *optimal cutting plane* (OCP for short). The cutting planes falling into the ε-region of an OCP, $\left|x_i^{uc} - x_i^{g}\right| \le \varepsilon$ (ε is the tolerance), are called the *ideal cutting plane*s (ICPs).

The cutting plane mechanism could reduce the problem to a one-variable problem (e.g. the concerned variable is x_i) and this reduced problem will be finely searched. To form a cutting plane, the unconcerned variables ($x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_d$) are treated as dummies by setting them at certain fixed values, which means the cutting plane is applied to intersect the hyper-surface. Take a two-variable problem $y = f\left(x_1, x_2\right)$ as

an example. Assume x_1 is the current concerned variable and the unconcerned variable x_2 is treated as a dummy by setting $x_2 = a$, the cutting plane is the gray area in Fig. 2 and the intercepted curve in the surface is shown in Fig. 3.

As shown in Fig. 3, $P\left(x_{1p}, y_p\right)$ is the optimum point of the intercepted curve. Obviously, only if the cutting plane is the OCP or an ICP, $P'\left(x_{1p}, a, y_p\right)$, the proto-point of P will be the desired global optimum or a satisfactory solution very close to the global optimum. Although the cutting plane in the example is not the OCP and thus P is not the projection of the global optimum, P is at least a solution near local optimum or global optimum.

Assume the probability that a randomly set value for a unconcerned variable x_i^{uc} is within the tolerance ε, denoted as $P_i = P(\left|x_i^{uc} - x_i^g\right| \le \varepsilon)$. Then the probability of getting an ICP is $P_{ICP} = \prod_i P_i$. Since P_i is inversely proportional to the range of value of x_i^{uc}, denoted as r_i, P_{ICP} would be rather small when r_i is large or the number of variables is large. The position of cutting plane is an important factor in obtaining a good projection of global optima. Therefore, the cutting plane needs to be moved adaptively. Hence, we adopt the concept of particle swarm optimization to assist in adjusting the cutting planes and set the cutting plane candidates as particles.

In each cutting plane, the optimum of the intercepted curve is to be searched by (1+1)-ES. PSO guides globally the cutting planes to the promising area and then local searching is conducted in the reduced search space (i.e. cutting planes). This kind of local via global search is believed to improve efficiency and accuracy. The details of the cutting plane moving mechanism will be described in Section 3.

2.4 Summary

In short,

1. The optima of a projected function in any dimension are the projections in that dimension of the corresponding global optima. This conclusion implies the feasibility of lowering the searching dimensionality of an optimization problem.
2. In the cutting plane mechanism, the closer to the OCP the cutting planes are, the more significant the found optima in the cutting planes. This analysis suggests the use of particle swarm optimization.
3. IES benefits from the combination of PSO and (1+1)-ES. PSO acts as a global guide while (1+1)-ES acts as a local fine tuner.

3. Incremental Evolution Strategy (IES)

Undoubtedly, searching in a lower dimensional space is easier and less time consuming. So, we can start the search for optima from only one concerned variable and approach the ultimate global optima by increments. If the information obtained before concerned-variable-increment can help evolution after the increment, such an incremental evolution can assure good quality of solution. The analysis in Section 2 shows the feasibility of such an incremental approach. This is the basic principle of IES, the details are shown below.

3.1 Architecture and Procedure of IES

IES divides the whole evolution into several phases. One more variable is considered in each phase until the global optima are found. Among all the phases, we call the first phase as the initial phase, the last phase as the ending phase, and those in between as intermediate phases. Each phase is composed of two steps. First, a population is evolved with regard to a certain concerned variable on some moving cutting plane adaptively, which is called SVE (Single-concerned-Variable Evolution).

Next, the better-performing individuals obtained from step one and the population obtained from the last phase are joined together in step two to form an initial population, to which PS-assisted MVE (Multi-concerned-Variable Evolution) is applied. The procedure is shown in Fig. 4, where S_i stands for SVE on variable i, i=1,2, … ,d. M_j stands for MVE with regard to from variable 1 to variable j+1, j=1,2, … , d-1, where d is the dimensionality of the original problem.

The algorithm works as follows (Assume there are d variables and N is the initial population size):

1. Set $k=1$, where k is the phase number. Generate a population and implement SVE (The details of SVE will be given later.) with regard to the first concerned variable. After that, m fittest individuals survive into MP_1 (MP_k represents the multi-concerned-variable population for phase k). Phase 1, namely the initial phase, then ends.
2. Set $k=k+1$. The next phase starts.
3. Generate a population and implement SVE with regard to the k-th concerned variable. After that, the m fittest individuals survive into SP_k (SP_k represents the single-concerned-variable population for phase k).
4. Generate the initial population for the multi-concerned-variable evolution in phase k, I_k, which is the result of integration operation on SP_k and MP_{k-1}. The details of integration operation will be given later.
5. If the size of I_k is larger than N, select the N fittest individuals. Then perform MVE (The details of MVE will be given later.) on I_k with the first to the k-th concerned variables. After the evolution, I_k evolves into MP_k. Phase k ends.
6. If none of the following stopping criteria is satisfied, go to 2.
   The stopping criteria are:

1) Stagnation Criteria: The improvement of the best individual in the past g_{strip} generations is less than a preset threshold ρ.

2) Generation Criteria: The generation number is larger than a preset threshold g_{thred}.

If anyone of them is satisfied, the whole evolution process finishes. The fittest individuals in the final population are the found optima.

3.2 Implementation of SVE and MVE

SVE aims to evolve the population with regard to only one concerned variable through continuously moving the cutting plane and finding optima on each cutting plane. To evolve the population on a certain cutting plane, (1+1) evolution strategy is used. To adaptively move cutting plane according to its previous performance in terms of the optima found on it, particle swarm optimization is applied.

- (1+1)-ES evolving

In a population, for each individual (chromosome), fix the cutting plane and evolve the concerned variable using mutation only. For global mutation we reset the concerned variable with a randomly generated value within the dynamic range of that variable at a small probability, while for local mutation we add the concerned variable with a small random value at a large probability. Each individual produces one offspring in a generation, and the better one among the parents and the offspring will be retained. This (1+1) evolution procedure is shown in Fig. 5.

As shown in the Fig. 5, the concerned variable-y plane is a cutting plane and the curve is the intercepted curve at the original hyper-surface intercepted by the cutting plane. The point K represents the projection of a chromosome on the cutting plane in the current population. During reproduction, the concerned variable would be mutated to a new random value at a small mutation probability p_{ms} and the

projection of the offspring on the current cutting plane is represented by the point R. Since the objective value of the offspring is less than that of the parent (minimization problem), the parent is replaced by the offspring. In the next generation, the concerned variable would be mutated to a new adjacent value at a large probability p_{ml} and the projection of the offspring on the current cutting plane is represented by the point B. Since the objective value of the offspring is again less than that of the parent, the parent is replaced by the offspring. Based on the stagnation criterion, the optima of the cutting plane could be found. There is one optimum in this cutting plane, represented by the point P, which is the target to be found using (1+1)-ES evolving on this cutting plane.

- Particle Swarm-assisted Moving Cutting Plane

PSO is a method that pulls the initial population of particles to the optimal solutions, which is what we need: moving the cutting planes towards the OCP. The detailed moving steps are described as follows:

a. In the initial phase, the concerned variable is x_1, there is one cutting plane corresponding to each chromosome that has different $x_2, x_3, \ldots x_d$ from the others. The chromosome is shown in Fig. 6.

b. To find the optimal x_1, represented by x_1* as shown in Fig. 7, for each cutting plane byusing (1+1)-ES evolving.

c. To compare all the chromosomes, choose the one with the smallest objective value as the global_best P_{gb} and update the local_best P_{lb}^i ($i = 2, \ldots, d$) of each chromosome if its current objective value is the smallest it ever achieved. And then adjust the cutting plane according to the update rule of PSO. The adjustment of the jth unconcerned variable of the ith chromosome x_j^i at time k is described as follows:

$$\begin{cases} v_j^i(k+1) = w \cdot v_j^i(k) + c_1 \cdot rand(\ ) \cdot \left(P_{gb}(k) - x_j^i(k)\right) + c_2 \cdot rand(\ ) \cdot \left(P_{lb}^i(k) - x_j^i(k)\right) \\ x_j^i(k+1) = x_j^i(k) + v_j^i(k+1) \end{cases}$$

$i = 1, 2, \ldots, M \ and \ j = 2, \ldots, d$

where M is the number of chromosomes in the current population and w, c_1 and c_2 are all constants in common PSO.

MVE is an extension of the cutting plane mechanism for SVE. The number of concerned variables is more than one and is equal to the sequence number of current phase. So we search the projection of global optima in cutting spaces with continuously incremented dimensions rather than in a cutting plane. The steps in MVE are similar to those in SVE. Firstly, use (1+1)-ES with regard to the current concerned variables to find the projections of global optima in a cutting space of certain dimensionality. Secondly, with the assistance from PSO, move the cutting space according to its previous performance in terms of minimal objective value found in it and the guide information from the best cutting space ever achieved. Continuously perform these two steps until stagnation in terms of the objective value of the best individuals.

3.3 Operation of Integration

The motivation of integration is to retain all the useful information and combine them to create some potential solutions. The procedure of integrating MP_{k-1} with SP_k into I_k, which is the initial population of k-th MVE M_k, is illustrated in Fig. 8. As shown in Fig. 8, all the chromosomes in both MP_{k-1} and SP_k are copied into I_k. Besides, for each chromosome in MP_{k-1}, its concerned variables are retained (from x_1 to x_{k-1}), then get a value for x_k from each chromosome in SP_k, lastly fill up the chromosome from x_{k+1} to x_d respectively with the corresponding parts of the two

chromosomes under integration. The found optimal value is marked by '*'. Please note that when $k = d$ the integration operation will be simply copying MP_{d-1} and SP_d into I_d.

4. Experiments and Results

4.1 Performance Evaluation Metrics

For parameter optimization problems, namely, both the independent variables and the objective function are scalar quantities, the numerical values of the variables are sought for which the value of the objective function is an optimum [1]. Corresponding to this goal, the following metrics were used:

1) y is the optimal objective value obtained.

2) γ is the Euclidean distance between the found optimum and the true global optimum.

Besides, the standard deviations of the above metrics are given as σ_y, σ_γ.

4.2 Experimental Scheme

The proposed algorithm has been implemented to solve several benchmark problems, which are commonly used test functions with different characteristics and degrees of difficulties. The results are the average of 50 runs with different seeds. In each run a different random sequence is used by setting the initial seed from 1 to 50.

The results are compared to the results of improved normal GA (INGA), PSO and SADE_CERAF. INGA improves the normal GA by dynamically changing the crossover and mutation probabilities [22]. There are two types of adaptation procedure as shown in Fig. 9, where gdm represents the ratio between the mean and the maximum values of the fitness function at each generation, called genetic diversity. pc and pm are respectively the probability of crossover and mutation.

One is based on linear interpolation (LI), while the other one is based on a measure of genetic diversity exterior to some limits (EL). The PSO program and the SADE program were downloaded respectively from their creator's homepages. And the SADE program was combined with the CERAF technology based on the concept from [21].

All the stopping criteria used in our experiments are stagnation criteria. Since the parameters for evolutionary algorithms are always set according to experiences, the parameters in our experiments were chosen according to a preprocessing procedure, in which different values were set to each parameter and those resulting in better results were chosen as shown in Table 1. All the experiments were done on a Pentium M 1.5GHz PC with 512MB memory.

4.3 Experimental Results

To ensure fairness of comparison, researchers usually use equal generation number/evaluation number of objective function/running time for comparison. In our experiments, the semantics of one generation for the algorithms in comparison is different. And since the object of an optimization algorithm is to find solutions as soon as possible, we use time limit, which means the evolutions for solving a problem by each algorithm are confined with the same period of time.

Since optimization algorithms often have their own strategies for achieving good performance, there would be some special parameters for each algorithm. With regard to the algorithms in comparison, the settings for their specific parameters follow the same settings as described in the original papers [22][2][21], as shown in Table 1.

4.3.1 **Problem 1**: Peaks Function (refer to APPENDIX)

In this 2-d problem, there is nonlinear interaction between the two variables and thus

the function is non-separable. Two OCPs of the Peak function and are shown in Fig. 10. And the true global minimum $\left(x_1^g, x_2^g, y^g\right)$ is $(0.23, -1.6250, -6.5511)$.

The time limit as the stopping criterion is 0.3s. The performance of the compared algorithms is shown in Table 2. As the results show, IES, PSO and SADE_CERAF can obtain the global solution for Problem 1 but INGA can not. Among these three algorithms, IES performs slightly better in the metric γ.

4.3.2 **Problem 2**: Rastrigin Function (refer to APPENDIX)

In this problem, the Rastrigin function is scalable and the interaction among variables is linear. The OCPs along all the dimensions are the same due to symmetry, as shown in Fig. 11. The true global minimum $\left(x_1^g, \ldots, x_d^g, y^g\right)$ for the scalable Rastrigin problem is $(0, \ldots, 0, 0)$.

In order to test the searching capacity of IES in higher dimensional searching spaces, the performance of IES and the other three algorithms on the Rastrigin function with the number of variable increased up to 30 is compared. Since the time consumed by IES would become long if all the phases are evolved in a high dimensional situation, a stopping criteria for each phase is imposed as described in 4.3. The average numbers of phases done when $d = 20$ and $d = 30$ over 50 runs are respectively 10.92 and 11.06. The time limit varies with the increase of dimensionality, respectively 0.3s, 2s, 6s, 15s, 20s. The results are shown in Table 3.

As the results in Problem 2 show, for almost all the dimensionalities IES performs better then the other three algorithms in metrics: y, σ_y, γ and σ_γ. With regard to these four metrics, the performance of IES, INGA and PSO gets worse with an increasing number of variables, while INGA achieves its best performance when $d = 10$.

4.3.3 **Problem 3**: Griewangk Function (refer to APPENDIX)

In this problem, the Griewangk function is scalable and the interactions among variables are nonlinear. So, this is a non-separable problem. And the true global minimum $\left(x_1^g,\ldots x_d^g, y^g\right)$ for the scalable Griewangk problem is also $(0,\ldots,0,0)$. According to [3], the Griewangk function has a flaw when scaled. The summation term of it induces a parabolic shape while the cosine function in the product term creates "waves" over the parabolic surface creating local optima. However, as the dimensionality of the search space is increased the contribution of the product term is reduced and the local optima become small. When the dimensionality is increased up to 10, the local optima almost disappear. So, we just test the performance of the algorithms on the Griewangk function with the number of variables increased from 2 to 6. Fig. 12 shows the OCPs of Griewangk function, when $d=2$ and $d=6$. The time limit varies with the increasing of dimensionality, respectively 0.3s, 1s, 1.5s, 2s, 3.5s. The results are shown in Table 4.

As the results in Problem 3 show, for almost all the dimensionalities IES performs better than the other three algorithms in metrics: y, σ_y, γ and σ_γ. With regard to these four metrics, the performance of IES, INGA and PSO gets worse with an increasing number of variables, while that of INGA gets better.

4.4 Analysis of Experimental Results

From the experiments on these three benchmark functions, we have an overall picture of IES. In summary, the results showed that:

1. IES generally outperforms the other three algorithms in the sense that the solutions found by IES are closer to the true optima and the minimal objective values found by IES are more optimal. And the chance of being trapped in local optima using IES is smaller than using the other three algorithms.

2. As the number of variables increases, not all the variables need to be evolved finely. If all the variables are evolved, which means the number of evolving phases is equal to the number of variables, the superiority of IES could be improved as the number of variables is increased as shown in Problem 3 (Fig. 13). Since the performance of INGA is much worse than the other three algorithms in this problem, its results are not presented in the figure to avoid affecting the scale. If a smaller number of evolving phases is implemented instead, the difference between the performance of IES and the other algorithms may be decreased, which can be observed in Problem 2. In any case the superiority of IES would not disappear, as shown in Fig. 14.

Given enough time, IES generally can help find solutions closer to the true optima with more optimal objective values. Especially, the results also suggested that with regard to high dimensionality problems on which normal algorithms could not give satisfactory performance, IES could perform better. In order to get some tradeoff between performance and time consumption, the number of phases evolved in IES should be less than the number of variables when the dimensionality is huge. The partially evolved IES still could obtain better solutions, given the comparison results in this paper.

5. Discussion

The reason why IES could have such a good performance is explained as follows:

1) PSO collaborates with ES in searching.

Let $f : R^d \to R$ be the objective function to be minimized. The simple (1+1)-ES can be modeled as a Markov process $(X_k)_{k\geq 0}$ such that [26]:

$$X_{k+1} = \begin{cases} X_k + l_k Z_k, if\ f(X_k + l_k Z_k) < f(X_k) \\ X_k, \qquad otherwise \end{cases}$$

where l_k is the step length. A state in the Markov process only depends on the state just before it. There is no memory and dependence on the states further before. That is to say, (1+1) ES is to some extent a local search algorithm.

In contrast, the process $(X_k)_{k\geq 0}$, generated by the PSO can be modeled as follows:

$$X_{k+1}=\begin{cases} X_k + & r2_{k+1}(GB_k - X_k), \text{if } f(GB_k)\leq f(X_k)\leq f(LB_k) \\ X_k, & \text{if } f(X_k)\leq f(GB_k) \\ X_k + r1_{k+1}(LB_k - X_k) + r2_{k+1}(GB_k - X_k), & \text{otherwise} \end{cases}$$

where $r1_{k+1}$ and $r2_{k+1}$ are the random velocity acceleration factors, GB_k is the global best chromosome out of all the chromosomes found in the past k generations and LB_k is the best point on the trace of a chromosome in the past k generations. The search performed by PSO is a non-Markov process, which depends on the memory of previous traces of the chromosomes. So PSO could specialize in global search.

To integrate these two algorithms, the global one should play the role of directing the search and the local one finetuning the search. Following this guideline, in IES, PSO is used to adjust the cutting plane/hyper-plane and the (1+1) ES is for finely searching on the cutting planes. In this way, the success probability of finding the global optima is increased as expected in Section 2.

2) Effective information accumulation by incremental evolution

- Contribution of SVEs

Assumptions:

a) In total n SVEs are conducted for a d-dimensional problem, $1 < n \leq d$.

b) The success probability of finding the global optima by each SVE is $p_1, p_2, \ldots, p_n$ respectively, and the minimum success probability is $p_{\min} = \min\{p_1, p_2, \ldots, p_i\}$.

c) The success probabilities of single PSO and single (1+1)-ES are respectively p_{PSO} and p_{11ES}, and the larger one between these two is $p = \max\{p_{PSO}, p_{11ES}\}$.

Let S denote the number of successful SVEs that find the global optimum. Then, the possibility of finding the global optimum in the n SVEs should be:

$$p_{SVEs} = P(S \geq 1) = 1 - P(S = 0) = 1 - \prod_{i=1}^{n}(1 - p_i)$$

Since usually $p_i \ll 1$,

$$p_{SVEs} = 1 - \prod_{i=1}^{n}(1 - p_i) \approx 1 - \left(1 - \sum_{i=1}^{n} p_i\right) = \sum_{i=1}^{n} p_i$$

The probability of finding the global optimum can be increased by using n SVEs.

According to the discussion in 1), in each SVE there is a dimension to be finely searched by the cooperation of (1+1)-ES and PSO. This combination of global and local search could result in better performance. Thus, the success probability of any SVE is believed to be larger than (at least equal to) the success probability of solely using (1+1)-ES or PSO, which means $p_{\min} \geq p$. Then,

$$p_{SVEs} = 1 - \prod_{i=1}^{n}(1 - p_i) \geq 1 - (1 - p_{\min})^n \geq 1 - (1 - p)^n$$

Since the success probability for any difficult problem by using (1+1)-ES or PSO is very small, which can be denoted as $p \ll 1$, then

$$p_{SVEs} \geq 1 - (1 - p)^n \approx 1 - (1 - np) = np .$$

Consequently for a difficult problem, the SVEs could make contribution in the sense that the probability of finding global optimum using n SVEs will be more than n times greater than using single (1+1)-ES or PSO.

- Role of MVEs

The searching in MVEs is focused near the solutions obtained from SVEs. Since the solutions derived from SVEs are in the adjacent region of either some local optima or global optimum, searching around them can further approach the local or global optimum. And the multi-SVE mechanism increases the possibility of approaching the adjacent region of the global optimum, as discussed above.

This can also be explained using the schema theorem and building block hypothesis [5]. A schema is a similarity template describing a subset of strings with similarities at certain string positions. It is postulated that an individual's high fitness is due to the fact that it contains good schemata. Short and high-performance schemata are combined to form building blocks with higher performance expected. Building blocks are propagated from generation to generation, which leads to a keystone of the GA approach. Research on GA has proved that it is beneficial to preserve building blocks during the evolution process. MVEs inherit the old chromosomes from SVEs and the previous MVEs, where the building blocks likely reside. The integration of these building blocks into the initial population provides a solid foundation for the following evolutions.

6. CONCLUSIONS

This paper first analyzed the effect of taking orthographic projection on objective functions. A conclusion was drawn which stated that the minima of the boundary function of orthographic projection of the hyper-surface corresponding to the original function are projections of the global optima. The cutting plane mechanism was proposed as an extension of this orthographic projection. We discussed and motivated the validity of optimizing projections of the objective function in lower dimensional spaces, thereby the feasibility of incremental evolution.

The incremental evolution strategy (IES) is proposed as a continuous incremental optimizer. Rather than evolving parameters in batch as done by normal optimization algorithms, IES finely evolves parameters one after another. Particle swarm optimization helps adjust the cutting planes, while (1+1)-ES helps find optima in the cutting planes/hyper-planes.

Experiments on three benchmark functions were done and the performance of IES was compared with other evolutionary algorithms, namely INGA, PSO and SADE_CERAF. The results showed that IES outperformed them such that it could find solutions with higher qualities both in the input space and the output space. Some explanation was given for the better performance obtained by IES.

In future, we will continue the study in two aspects: 1) The success of IES exhibits its potential to solve problems with a dynamic variable set as expected in Section 1. We plan to apply incremental evolution to such dynamic function optimization problems. 2) We would also like to see if we can apply the proposed algorithm in other domains, which have dynamic fitness landscapes with a changing number of variables, such as clustering problems without prior knowledge of the number of clusters.

TABLES

Table 1 Parameter settings

<table>
<tr><td colspan="3">Common Settings</td><td>Swarm size: 50</td></tr>
<tr><td rowspan="4">Specific Settings</td><td rowspan="2">INGA[1)]</td><td>LI</td><td>$gdm\_\min = 0$, $gdm\_\max = 1.0$
crossover parameters:
$pc\_\min = 0.5$, $pc\_\max = 1.0$
mutation parameters:
$pm\_\min = 0.025$, $pm\_\max = 0.25$</td></tr>
<tr><td>EL</td><td>$gdm\_\min = 0.005$, $gdm\_\max = 0.15$, $km = kc = 1.1$
crossover parameters:
$pc\_\min = 0.5$, $pc\_\max = 1.0$
mutation parameters:
$pm\_\min = 0.001$, $pm\_\max = 0.25$</td></tr>
<tr><td colspan="2">PSO[2)]</td><td>inertia weight: $w = 0.729$
acceleration constants: $c_1 = c_2 = 1.49445$</td></tr>
<tr><td colspan="2">SADE_CERAF[3)]</td><td>crossover rate: $pc = 0.2$
mutation parameters: $pm = 0.5$, $mf = 0.25$
radioactivity factor: $rf = 0.25$</td></tr>
<tr><td colspan="2">IES[4)]</td><td>mutation parameters: $pgm\_s = 0.2$, $pgm\_m = 1/n$, $plm = 0.8$
inertia weight: $w = 0.729$
acceleration constants: $c_1 = c_2 = 1.49445$
inheritance parameters: $in\_s = 6$, $in\_m = 20$</td></tr>
</table>

Remarks:

1) Refer to Fig. 7 with regard to the meanings of the parameters.

3) pm is the mutation rate. mf is the ratio of the mutation range to the corresponding domain range. rf is the ratio of the radioactivity area to the corresponding domain.

4) pgm_s and pgm_m are the mutation rates for global search of SVEs and MVEs, respectively. plm is the mutation rate for local search. in_s and in_m are the numbers of solutions inherited from SVE and MVE, respectively.

The stopping criteria for searching in the cutting plane: the enhancement of the fittest individual in the population is less than 0.1% over the last 10 generations or the generation number is more than 1000.

Similarly, the stopping criteria for SVEs and intermediate MVEs: the enhancement of the fittest individual is less than 0.1% in the last 10 cutting planes or the time of moving cutting plane is more than 1000.

In addition, if the variable number is larger than 10, the number of phases could be less than the number of variables. The stopping criteria: the enhancement of the fittest individual is less than 0.1% in the last 5 phases or all the phases are finished.

Table 2 Performance comparison on Peaks function

	IES	INGA		PSO	SADE_CERAF
		LI	EL		
y	-6.5511	1.1641	-1.1534	-6.5511	-6.5511
(σy)	(1.95E-07)	(1.1967)	(1.2029)	(2.72E-07)	(2.72E-07)
γ	0.0012	1.5207	1.5422	0.0022	0.0018
$(\sigma\gamma)$	(7.08E-11)	(0.7190)	(0.7290)	(7.8E-11)	(7.06E-11)

Legends: y : The optimal objective value obtained; σ_y : The standard deviation of y ;
γ : The Euclidean distance between the found optima and the true global optima;
σ_γ : The standard deviation of γ ;
t : The elapsed time of the whole evolving process evaluated in seconds;
σ_t : The standard deviation of t .

Table 3 Performance comparison on Rastrigin function

		IES	INGA		PSO	SADE_CERAF
			LI	EL		
y (Δy)	d = 2	0.0199	26.0220	25.6106	0.0254	0.2404
		(0.0141)	(9.4635)	(9.1913)	(0.0092)	(0.2863)
	d = 5	0.3869	3.5152	9.1151	0.5851	3.3630
		(0.5410)	(1.9426)	(4.1200)	(0.5884)	(1.7967)
	d = 10	0.9950	2.0111	4.1545	1.6989	11.9992
		(0.8039)	(1.1732)	(2.1026)	(0.9039)	(5.2702)
	d = 20	2.7110	7.9696	18.6603	19.9083	25.5107
		(1.5592)	(7.1554)	(5.0669)	(5.2019)	(10.2846)
	d = 30	21.6967	32.7483	40.0999	41.3081	38.2263
		(11.1530)	(23.9350)	(7.2578)	(11.0087)	(12.4872)
γ $(\Delta\gamma)$	d = 2	1.2E-05	4.1473	4.2066	0.0661	0.1990
		(3.44E-13)	(1.4125)	(1.4078)	(0.1948)	(0.1407)
	d = 5	0.2971	1.3080	2.5010	0.6405	1.7556
		(0.4480)	(0.6668)	(0.7167)	(0.3521)	(0.5190)
	d = 10	0.3326	0.4871	1.8143	0.9949	3.3732
		(0.4807)	(0.5742)	(0.5979)	(0.3810)	(0.7562)
	d = 20	0.4198	1.5798	3.8619	4.1970	4.9394
		(0.6088)	(1.4707)	(0.6756)	(0.9215)	(1.0018)
	d = 30	2.4668	3.4726	5.5345	5.1196	6.0925
		(0.5416)	(2.1372)	(1.1122)	(0.9002)	(0.9661)

Legends: Refer to Table 2.

Table 4 Performance comparison on Griewangk function

		IES	INGA		PSO	SADE_CERAF
			LI	EL		
y (Δy)	d = 2	2.5E-05	34.8332	34.9743	0.0001	0.0007
		(1.8E-11)	(23.0031)	(23.2077)	(0)	(0.0022)
	d = 3	0.0024	19.0690	27.4298	0.0099	0.0118
		(0.0015)	(14.3134)	(19.3736)	(0.0011)	(0.0106)
	d = 4	0.0128	2.3957	9.3614	0.0321	0.0323
		(0.0086)	(0.7123)	(12.2083)	(0.0095)	(0.0244)
	d = 5	0.0481	1.2291	2.2387	0.1275	0.0724
		(0.0257)	(0.3048)	(1.9611)	(0.0228)	(0.0580)
	d = 6	0.0645	0.9968	1.3770	0.2862	0.1170
		(0.0304)	(0.2725)	(0.3134)	(0.0297)	(0.0681)
γ $(\Delta\gamma)$	d = 2	0.0657	340.6121	341.2636	0.3	0.5437
		(0.0004)	(140.8748)	(141.6924)	(0.0085)	(1.6476)
	d = 3	5.0714	252.7572	304.2037	6.2754	6.0395
		(2.9330)	(93.3550)	(116.0900)	(3.0002)	(3.3202)
	d = 4	9.0965	87.0347	165.6331	9.9067	10.4361
		(3.6979)	(13.8549)	(83.2273)	(3.9980)	(4.5472)
	d = 5	10.8636	61.8433	80.2798	20.75909	15.9084
		(3.4865)	(7.6757)	(26.4945)	(4.0392)	(6.0774)
	d = 6	13.2547	53.1455	61.0173	29.1942	20.8049
		(3.6254)	(8.1528)	(9.6957)	(6.2351)	(5.9272)

Legends: refer to Table 2.

LIST OF FIGURES

FIGURES

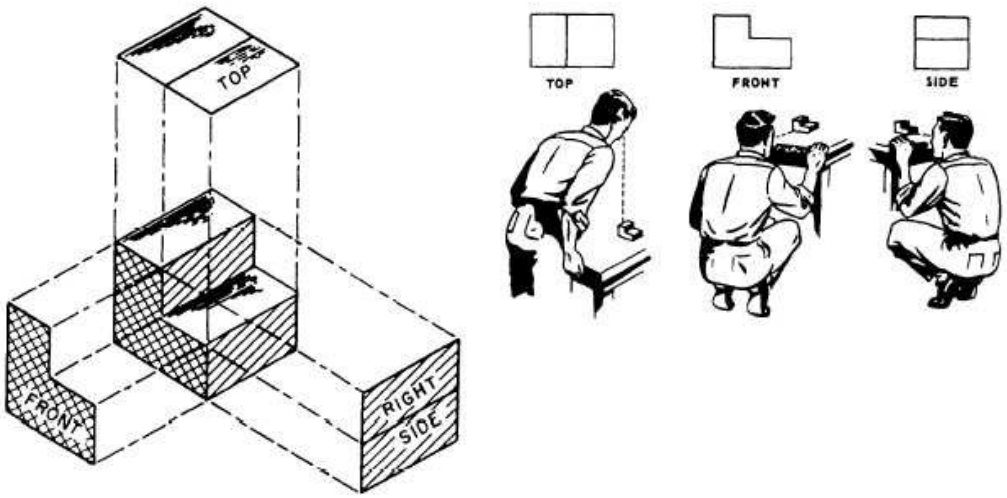


Fig. 1 A three-view orthographic projection

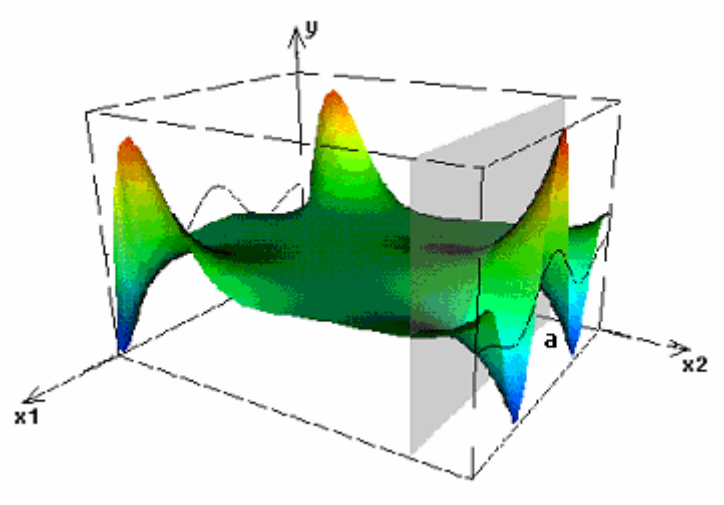


Fig. 2 Cutting plane for a two-variable problem

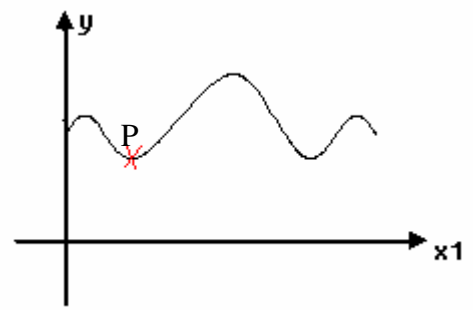


Fig. 3 Intercepted curve in the surface

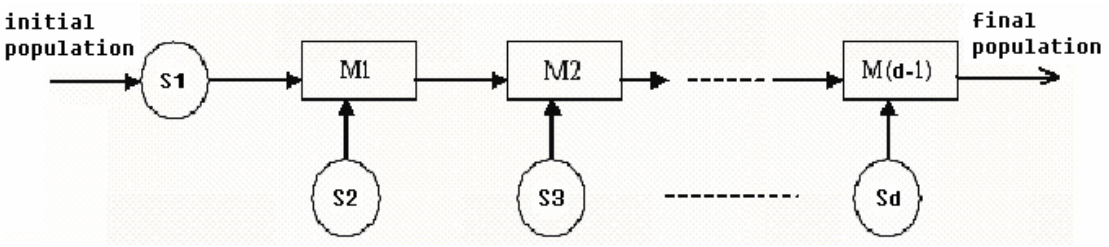


Fig. 4 IES procedure

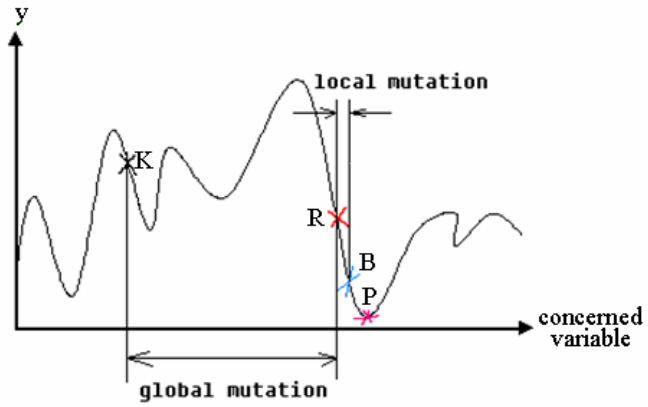


Fig. 5 Illustration of (1+1)-ES in SVE

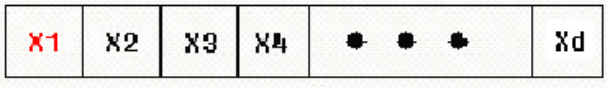


Fig. 6 Chromosome in SVE_1

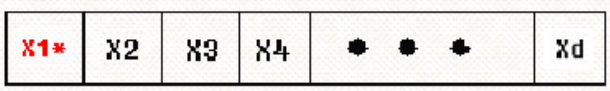


Fig. 7 Solution found in a cutting plane of SVE_1

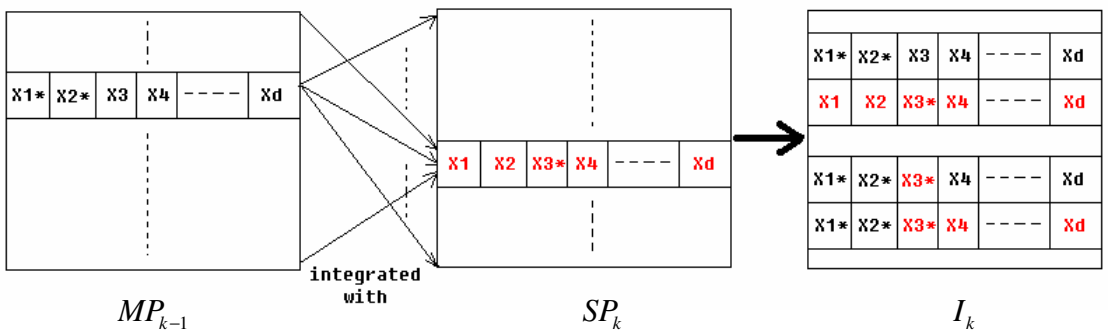


Fig. 8 Integration operation (assume $k = 3$)

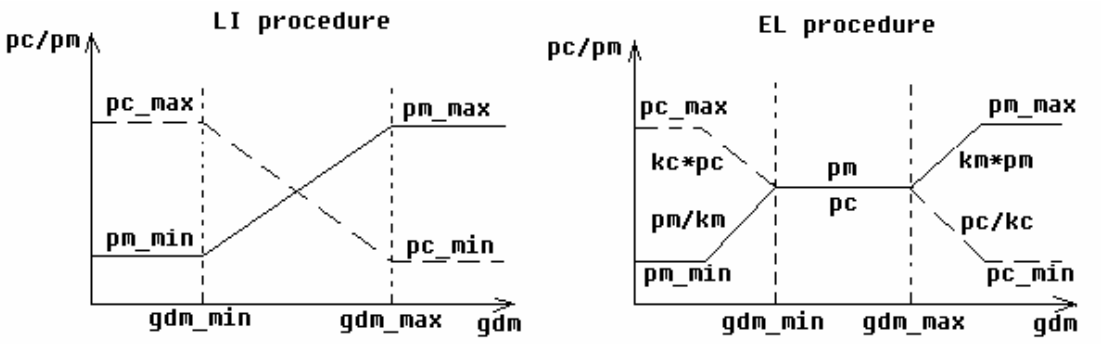
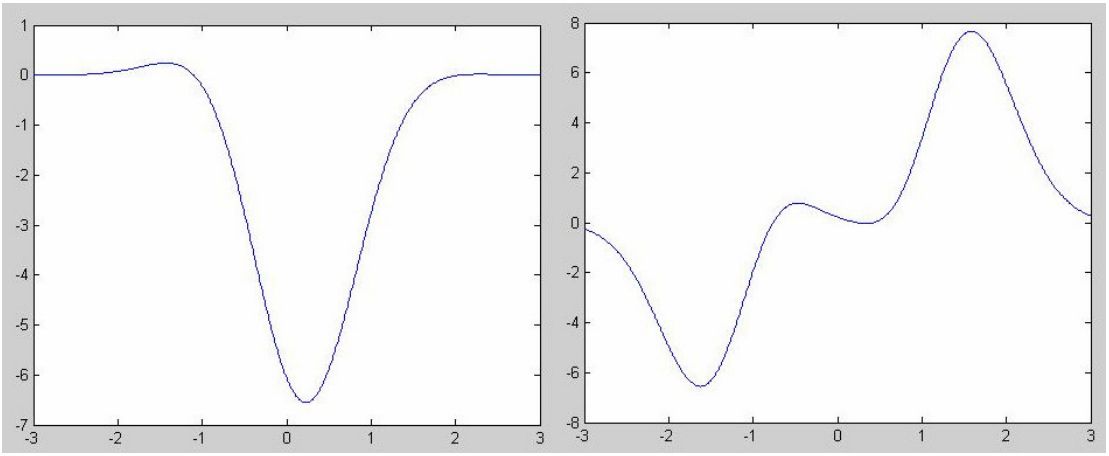


Fig. 9 Two types of dynamic adaptation



$(x_2 = -1.625)$ $(x_1 = 0.23)$

Fig. 10 Peaks function and two OCPs

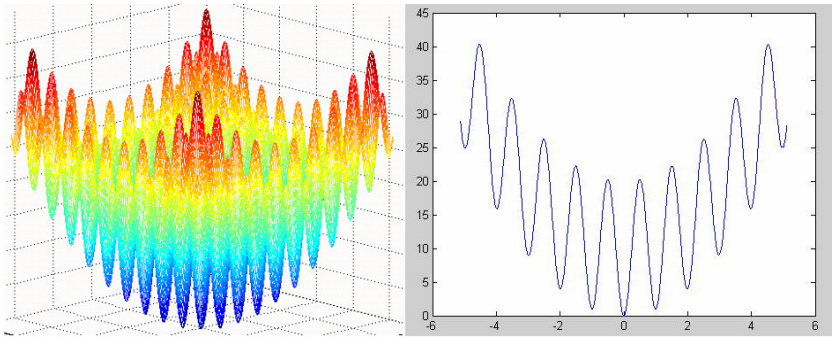
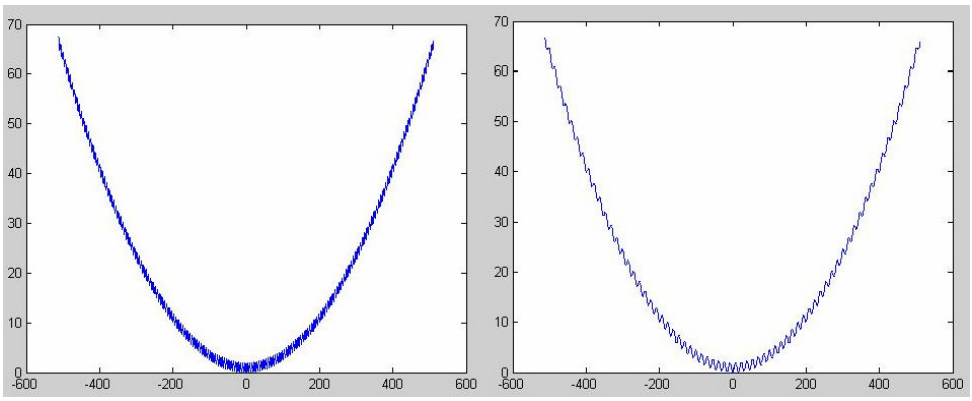


Fig. 11 Hyper-surface of Rastrigin function ($d = 2$) and OCP for all dimensions



OCP$(d=2)$ OCP $(d=6)$

Fig. 12 Hyper-surface of Griewangk function ($d=2$) and OCPs along $x_1, \ldots, x_6$

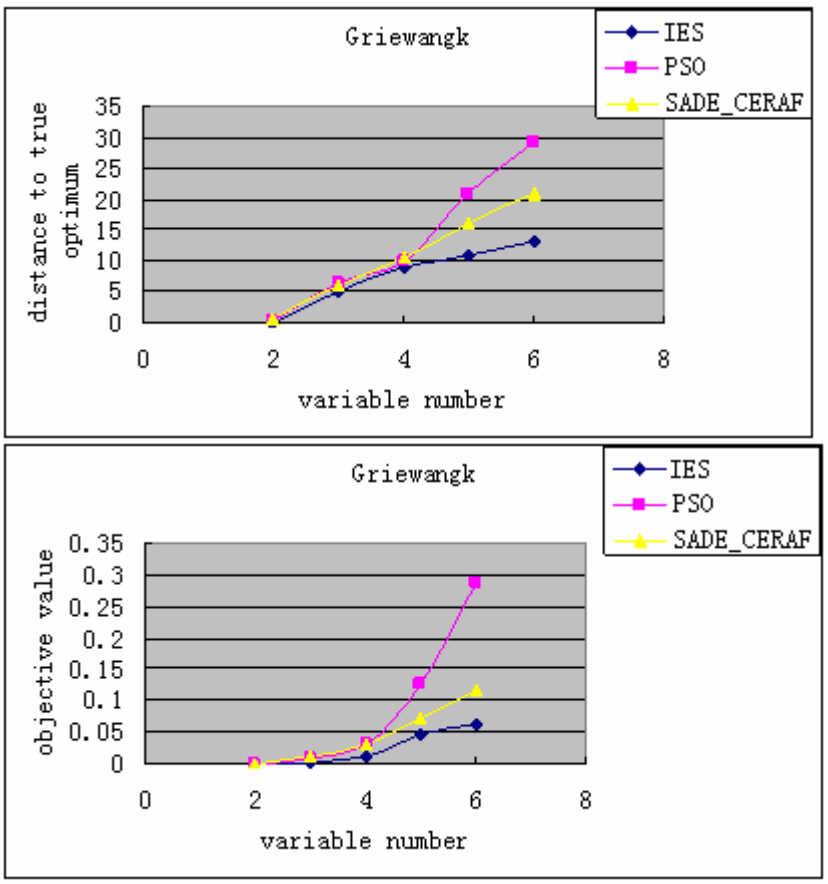


Fig. 13 Superiority of IES on Griewangk problem (Problem 3)

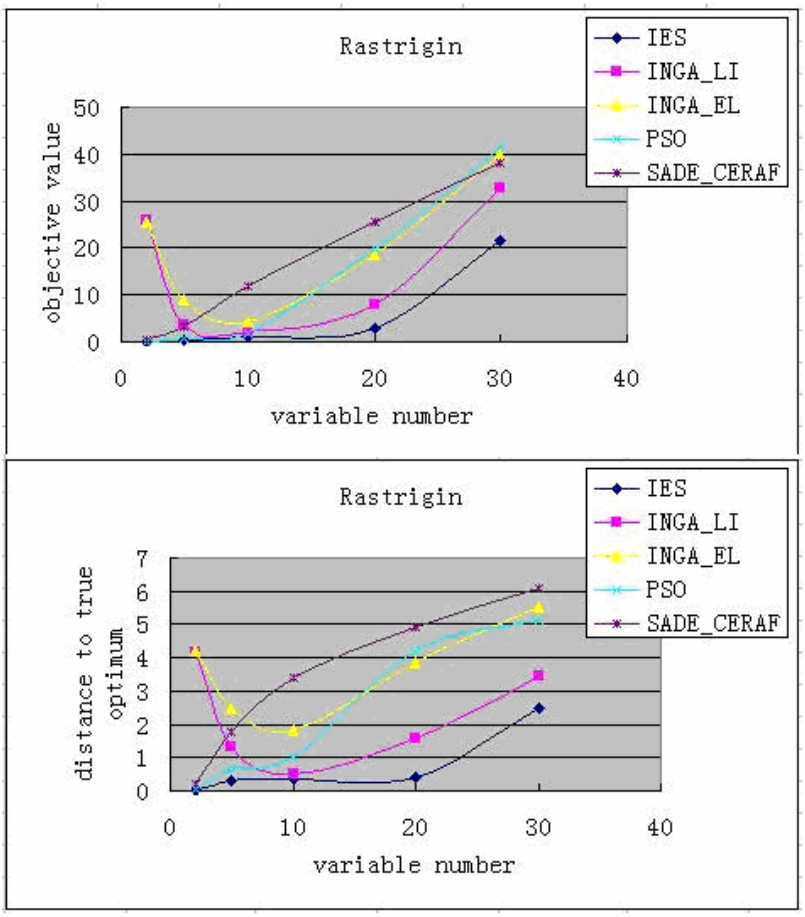


Fig. 14 Superiority of IES on Rastrigin problem (Problem 2)

APPENDIX

The Peaks function is:

$$f(x_1, x_2) = 3*(1-x_1)^2 * e^{-x_1^2-(x_2+1)^2} - \left[10*\left(x_1/5 - x_1^3 - x_2^5\right)*e^{-x_1^2-x_2^2} + 1/3*e^{-(x_1+1)^2-x_2^2}\right]$$

$$x_1, x_2 \in [-3.0, 3.0]$$

The Rastrigin function is:

$$f\left(x_i|_{i=1,\ldots,d}\right) = 10*d + \sum_{i=1}^{d}\left[x_i^2 - 10*\cos(2*\pi*x_i)\right], \quad x_i \in [-5.12, 5.11]$$

The Griewangk function is:

$$f\left(x_i|_{i=1,\ldots,d}\right) = 1 + \sum_{i=1}^{d}\frac{x_i^2}{4000} - \prod_{i=1}^{d}\left(\cos\left(x_i/\sqrt{i}\right)\right), \quad x_i \in [-512, 511]$$

REFERENCES

[1] H. P. Schwefel, Evolution and optimum seeking. New York: John Wiley & Sons, 1995.

[2] R. C. Eberhart and Yuhui Shi, "Particle swarm optimization: developments, applications and resources," Proceedings of the 2001 Congress on Evolutionary Computation, vol.1: 81-86, 2001.

[3] H. Muhlenbein and D. Schlierkamp-Voosen. "Predictive models for the breeder genetic algorithm," Journal of evolutionary computation, 1(1): 25-49, 1993.

[4] A.E. Eiben and T. Back, "Empirical Investigation of Multiparent Recombination Operators in Evolution Strategies," Journal of Evolutionary Computation, 5(3): 345-365, 1997.

[5] John Holland, Adaptation in Natural and Artificial Systems. University of Michigan Press, 1975.

[6] David Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA, 1989.

[7] Hans-Paul Schwefel, Numerical Optimization of Computer Models. Wiley, 1981.

[8] Hans-Paul Schwefel, Evolution and Optimum Seeking. Wiley, 1995.

[9] T. Back, Evolutionary Algorithms in Theory and Practice. Oxford University Press, 1996.

[10] D. Whitley, "An Overview of Evolutionary Algorithms: Practical Issues and Common Pitfalls," Journal of Information and Software Technology 43:817-831, 2001.

[11] J. Kennedy and R.C. Eberhart, “Particle swarm optimization,” IEEE Int. Conf. on Neural Networks, 1942-1948, 1995.

[12] Y.H. Shi, R.C. Eberhart, “Fuzzy adaptive particle swarm optimization,” IEEE Int. Conf. on Evolutionary Computation, 101-106, 2001.

[13] P.J. Angeline, “Evolutionary optimization versus particle swarm optimization: philosophy and performance difference,” Annual Conf. on Evolutionary Programming, 601-610, 1998.

[14] J. Kennedy, “Bare bones particle swarms,” Intelligence Symposium, 80-87, 2003.

[15] X.F. Xie, W.J. Zhang, Z.L. Yang, “A dissipative particle swarm optimization,” Congress on Evolutionary Computation, 1456-1461, 2002.

[16] M. Clerc, J. Kennedy, “The particle swarm - explosion, stability, and convergence in a multidimensional complex space,” IEEE Trans. on Evolutionary Computation, 6(1): 58-73, 2002.

[17] T. I. Cristian, “The particle swarm optimization algorithm: convergence analysis and parameter selection,” Information Processing Letters, 85(6): 317–325, 2003.

[18] J. Kennedy, “The particle swarm: social adaptation of knowledge,” IEEE Int. Conf. on Evolutionary Computation, 303-308, 1997.

[19] J. Robinson, S. Sinton and Y.R. Samii, “Particle Swarm, Genetic Algorithm, and Their Hybrids: Optimization of a Profiled Corrugated Horn Antenna,” IEEE Antennas and Propagation Society International Symposium and URSI National Radio Science Meeting, San Antonio, TX. 2002.

[20] R. Storn, “On the use of differential evolution for function optimization,” In:

NAPHIS, 1996.

[21] Ondrej Hrstka and Anna Kucerova, “Improvements of real coded genetic algorithms based on differential operators preventing premature convergence,” Advances in Engineering Software 35, pp, 237-246, 2004.

[22] J. A. Vasconcelos, J. A. Ramirez, R. H. C. Takahashi and R. R. Saldanha, “Improvements in Genetic Algorithms,” IEEE TRANSACTIONS ON MAGNETICS, vol. 37, no. 5, Sep. 2001.

[23] P. J. Angeline, “Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences,” in Proc. Evol. Prog. VII, V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, Eds. Berlin, Germany: Springer-Verlag, pp. 601-610, 1998.

[24] Gurney, K. An Introduction to Neural Networks, London: Routledge, 1997.

[25] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” Proc. IEEE Int. Conf. on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, pp. 1942-1948, 1995.

[26] C. W. Ahn and R. S. Ramakrishna, “Elitism-Based Compact Genetic Algorithms,” IEEE Trans. on Evolutionary Computation, Vol. 7, No. 4, August 2003.

[27] Sheng-Uei Guan and Shanchun Li, "Incremental Learning with Respect to New Incoming Input Attributes," Neural Processing Letters, 241-260, Vol. 14, Issue 3, Dec. 2001.

[28] Sheng-Uei Guan and Jun Liu, "Incremental Ordered Neural Network Training," 137-172, Vol. 12, No. 3, Journal of Intelligent Systems, 2002.

[29] Sheng-Uei Guan and Peng Li, "A Hierarchical Incremental Learning Approach to Task Decomposition," 201-226, Vol. 12, No. 3, Journal of Intelligent Systems, 2002

[30] Sheng-Uei Guan and Fangming Zhu, "Incremental Learning of Collaborative Classifier Agents with New Class Acquisition – An Incremental Genetic Algorithm Approach," 1173-1193, Vol. 18, No. 11, International Journal of Intelligent Systems, Nov. 2003.

[31] Sheng-Uei Guan and Jun Liu, "Incremental Neural Network Training with an Increasing Input Dimension," 45-70, Vol. 13, No. 1, Journal of Intelligent Systems, 2004.

[32] Sheng-Uei Guan and Peng Li, "Incremental Learning in Terms of Output Attributes," 95-122, Vol. 13, No. 2, Journal of Intelligent Systems, 2004.

[33] Qian Chen and Sheng-Uei Guan, "Incremental Multiple Objective Genetic Algorithms," 1325-1334, Vol. 34, No. 3, IEEE Transactions on Systems, Man and Cybernetics Part B, June 2004.